Single Document Storage Tool (SDS_tool) Maintenance, Installation and Configuration Guide

Release 1.0

May 2007

Notices

This document contains information proprietary to FileNet Corporation (FileNet). Due to continuing product development, product specifications and capabilities are subject to change without notice. You may not disclose or use any proprietary information or reproduce or transmit any part of this document in any form or by any means, electronic or mechanical, for any purpose, without written permission from FileNet.

FileNet has made every effort to keep the information in this document current and accurate as of the date of publication or revision. However, FileNet does not guarantee or imply that this document is error free or accurate with regard to any particular specification. In no event will FileNet be liable for direct, indirect, special incidental, or consequential damages resulting from any defect in the documentation, even if advised of the possibility of such damages. No FileNet agent, dealer, or employee is authorized to make any modification, extension, or addition to the above statements.

FileNet may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Furnishing this document does not provide any license to these patents, trademarks, copyrights, or other intellectual property.

U.S. Patents Disclosure

This product incorporates technology covered by one or more of the following patents:

U.S. Patent Numbers: 6,094,505; 5,768,416; 5,625,465; 5,369,508; 5,258,855

FileNet Corporation
3565 Harbor Boulevard
Costa Mesa, California 92626
800.FILENET (345.3638)
Outside the U.S., call:
1.714.327.3400
www.filenet.com

Contents

About This Manual

This document, the *Image Services Single Document Storage Tool (SDS_Tool) Maintenance, Installation and Configuration Guide,* contains information for installing and configuring the SDS_tool in conjunction with EMC Centera, Network Appliance SnapLock, and IBM TotalStorage DR550 storage solutions. It is intended to be read by people who are familiar with the UNIX and/or Windows Server Operating Systems, as well as with the FileNet Image Services software.

Note Throughout this document, the Single Document Storage Tool product is referred to as the SDS_tool and the IBM Tivoli Storage Manager software is referred to as IBM TotalStorage DR550.

Conventions Used in this Manual

The following paragraphs discuss the ways in which we call your attention to information throughout this document.

File Paths

Since this manual is used for all platforms, examples of file path designations, where used, are given for both UNIX and Windows platforms. For example:

/fnsw/local/sd (UNIX)

<drive>:\fnsw_loc\sd (Windows)

Console Displays

Information you see displayed at your console is shown in this document in the following manner:

```
Surface '3176' : 1 document processed
Local doc_id = '2235007' Original doc_id = '2235007'
Original ssn = '8502'
Primary copy. No tranlog copy exists.
* document successfully deleted from databases. *
* Purging pages from disk... *
* This document has been successfully purged. *
```

Cautions, Notes, and Tips

Three message types call your attention to important information:

CAUTION Signals possible damaging consequences of an action, such as loss of data or time.

Note Draws your attention to essential information you should be sure to read.

Tip Introduces an idea that might make your work easier.

Command Syntax

Command syntax definitions are indented and in bold text:

ddexim -e > <filename>

Optional Parameters

Optional parameters and keywords are within square brackets:

ddexim [-e] > <filename>

Required Parameters

Parameters that require you to provide information are shown within angle brackets (< >).

For example, for the following command:

ddexim -e > <filename>

you must substitute the name of a file for the parameter in angle brackets, such as:

ddexim -e > myfile

FileNet Education

FileNet provides various forms of instruction. Please visit Global Learning Services on FileNet's Web site at: **www.filenet.com**.

Comments and Suggestions

FileNet invites all customers to communicate with the Documentation group on any question or comment related to FileNet manuals and on-line help. Send email to **docs@filenet.com**. We will make every effort to respond within one week. Your suggestions help us improve the products we deliver.

1
Getting Started

Introduction

This document describes installing, configuring, and using the Single Document Storage Tool (SDS_tool). It assumes familiarity with the UNIX™ or Windows™ operating systems, as well as the FileNet Image Services (IS) software.

SDS_tool operation is best understood within the context of the overall SDS process, especially Ingestion. This process is implemented in two phases: the Image Services committal process, and the Near Line Storage (NLS) archival process.

When documents are committed to Image Services, entries are created in the Index database and Document Locator table. The document content is stored in the page cache for retrieval by client applications.

NLS performs the archival process, which is initiated independently of the committal process. The process monitors the Image Services Index database on a periodic basis (typically every few minutes) for newly committed documents. It then archives the committed documents to a protected storage device.

When committed documents are archived by NLS, an archive object is built, which includes the document's content and metadata, and then transferred to the protected storage device (SDS device). Archive

objects transferred to the protected storage device are called “SDS objects.”

For file system-based SDS devices, the SDS object name is the document ID. For API-based devices, NLS receives an SDS object ID from the device, which is then written to the FNP_CLIPID field in the IS Index database. The FNP_ARCHIVE field is populated with the current date. For more information, refer to *Single Document Storage and Retrieval Procedures and Guidelines*.

At archival time, retention settings are applied to the SDS objects according to the values found in the Index database and defined for each Image Services document class. Image Services retention is defined as either chronological retention or event based retention (EBR). Chronological retention is based on the document’s “entry date” (the date it was committed), while event based retention is based on the document’s “close date” (as a result of a business event). In either case, a retention offset (in months) is applied to determine the document’s “delete date.” If the document’s retention is chronological, the delete date is determined at committal time and assigned immediately. If the document’s retention is event based, the delete date is not determined until the document is “closed.”

NLS calculates the SDS objects retention setting as follows:

At archival time, the retention settings applied by NLS to the SDS objects are controlled by the SDSRetention configuration variable in the nls.cfg configuration file. The SDSRetention configuration variable has four possible retention setting options:

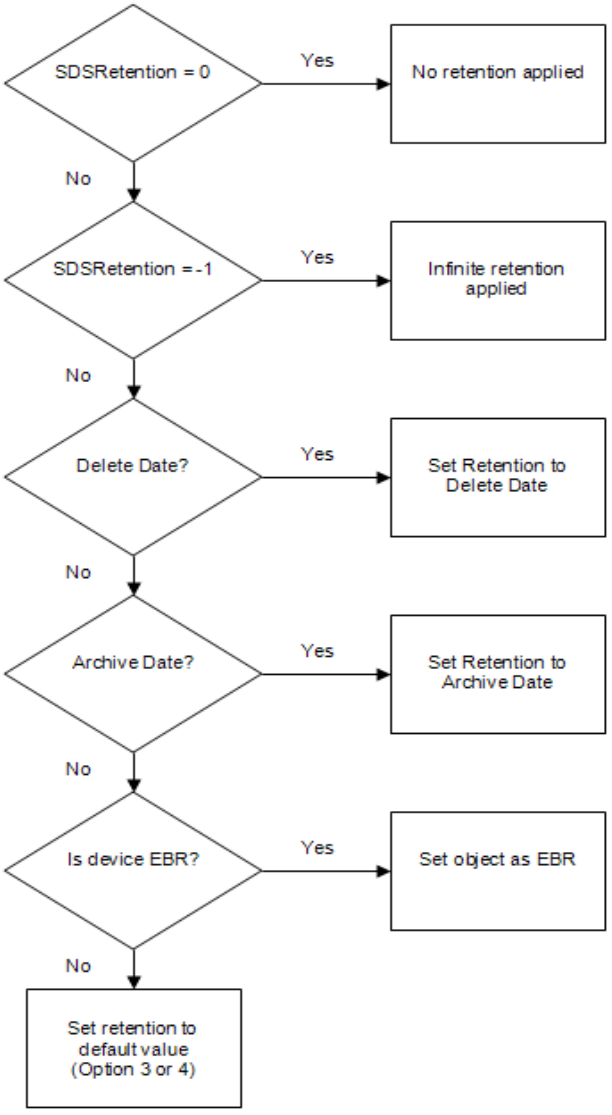
1. Zero (0): No retention is applied to the SDS objects.

2. Negative 1 (-1): The value indicates an infinite retention value is applied to the objects.

3. Greater that Zero (>0): The value is the retention value, in months.

4. Date Value (mm/dd/yy): The exact date when retention expires.

If options 1 or 2 are defined, the document's delete date and archive date in the Image Services Index database are ignored. Using options 1 or 2 overrides any chronological retention value stored in the Index Database for the document. If options 1 and 2 are not defined, NLS uses the document's delete date; if the delete date is not set, NLS uses the document's archive date. If the archive date is not set, NLS checks whether the SDS device supports EBR; if it does support EBR, the SDS objects are configured as EBR objects. If the device does not support EBR, options 3 or 4 are used as the default retention value.


SDSRetention = 0
Yes
No retention applied
No
SDSRetention = -1
Yes
Infinite retention applied
No
Delete Date?
Yes
Set Retention to Delete Date
No
Archive Date?
Yes
Set Retention to Archive Date
No
Is device EBR?
Yes
Set object as EBR
No
Set retention to default value (Option 3 or 4)

Caution NLS 2.0 implements Event-Based Retention (EBR) when used with an EBR-capable SDS device. As a result, when upgrading from an earlier NLS version and connecting to CentraStar 3.1 and later, NLS sets archive objects to EBR if no “delete date” or “archive date” are set and the SDSRetention parameter is set to a retention other than infinite or none. Review and update retention settings of document classes associated to SDS prior to upgrading to ensure retention is applied appropriately.

Note The License for Centera Storage and Retrieval (CSAR) to Image Services provides the customer with rights to store documents to one or several EMC’s Centera Basic, Governance, or Compliance Plus devices. This integration expands the range of fixed content storage options for FileNet customers, especially by providing fast access to massive amounts of magnetic WORM storage.

To take advantage of Event-Based Retention in Centera Governance and Compliance Plus editions, customers should make sure that they are appropriately licensed for it.

The SDS_tool application allows users to manipulate data objects stored on FileNet-supported SDS devices. The application presents a consistent command line interface that hides the differences among devices. SDS_tool has a set of basic but powerful commands that can be entered interactively or executed by means of a text file script. Resembling other FileNet Image Services tools, SDS_tool features online help and various options to facilitate logging, report creation and debugging.

An SDS object is a file containing a FileNet document, plus a header that holds the document's metadata. This ancillary data includes index entries, checksums, dates or other FileNet and SDS Device identifiers. The document is any FileNet-supported text or binary file, and must be archived to the SDS Device before SDS_tool can act upon it. Depending on the device type and how it is archived, the SDS object may or may not be present on a file system.

SDS_tool reports on, modifies, deletes and recovers SDS objects by interacting with the SDS Device and Image Services. An SDS_tool report includes FileNet IS data, SDS Device data or both; if both, it compares the data between the two sources for consistency. It can modify the expiration time, or retention, of an SDS Object, setting it, if it does not exist, or extending it, if does exist. If an object's retention has expired, SDS_tool can delete the object completely from the IS and SDS device. If an object's data is missing from the IS, SDS_tool can recover it from the SDS device, including its database records, and if an IS storage library exists, it can restore the document there as well. SDS_tool cannot re-archive an SDS object; that is, it cannot put a missing object back on the SDS device.

SDS_tool writes the results of its operations into a user-specified output file. The file's main purpose is to hold the report data for an object, but it can also contain the results of an object modification, deletion or recovery. For efficiency sake, it only writes a limited amount of data to the screen and writes the full details into the output file. Debugging information, status and error messages are sent to a separate, standard log file with a fixed naming convention and location.

SDS_tool is a unified set of interfaces that interact with the SDS devices. The tool manages documents from FileNet's Image Services on protected storage device libraries, and provides a unified interface with the same functions and syntax, regardless of the SDS device

accessed. SDS_tool functionality is the same for the UNIX or Windows platforms (see **Chapter 2, "Software Installation," on page 19** for details specific to your operating system.)

SDS_tool allows users to administer SDS repositories and objects in an integrated fashion, and to control these devices much like the familiar FileNet command line tools.

The FileNet System Consultant verifies prerequisites, reviews configuration options and assists users installing SDS_tool with the requirements before installing, configuring, and running SDS_tool.

SDS_tool uses an underlying software layer called the Hierarchical Storage Management Architecture (HSMA). The tool supports EMC's Centera™, IBM's TotalStorage DR550 ™ Storage Manager, Solaris SunSAR™, and Network Appliance's SnapLock™ products as storage peripherals. Other storage management products with a standard file system interface might work correctly with HSMA, but were not tested or validated.

The SDS_tool runs asynchronously. It does not store documents to an SDS device; NLS does. Instead, SDS_tool uses the Image Services document retention as the reference.

SDS_tool accesses the SDS device "BLOb," a file containing a single document and FileNet metadata. The SDS device repository contains enough data to recover FileNet documents to the most recent archived document version.

The SDS_tool process indexes and commits documents to Image Services with:

- f_entrydate = commit date

- f_retentoffset = as per doc class
- f_retentbase = as per doc class
- f_deletedate = f_entrydate + f_retentoffset if f_retentbase = “1” or else it is NULL.

NLS writes archive objects to the protected storage device. If the EBR parameter is available in the device and f_deletedate = NULL, SDS_tool writes EBR-enabled objects and sets on hold. If EBR is not available in the device, SDS_tool writes objects and applies chronological retention per f_deletedate if known, or else it applies the default.

Run SDS_tool to:

- Set the retention or activate EBR on archive objects.
- Extend retention consistently on an archive object.
- Delete a document from Image Services and the device.

Hardware and software requirements

The SDS_tool works in conjunction with NLS and is currently compiled for FileNet Image Services ToolKit (ISTK, previously called WorkFlo Application Library, or WAL) versions 3.6 and 4.0. Check the IS support matrix for operating system support relating to your IS and ISTK version. The IS software and hardware requirements, as well as the HSMA, to which you connect dictate hardware support requirements. Refer to IS documentation for FileNet hardware requirements.

When run on a UNIX system, the SDS_tool installation program detects the ISTK version and installs the correct SDS_tool version. No automatic version detection occurs when installing on a Windows

system. To determine the ISTK version, refer to the **Chapter 2, "Software Installation," on page 19** section. For specific questions, refer to the SDS_tool section in the most recent SDS solution product announcement.

The SDS_tool installation requires a minimum of 10 MB of hard disk space. The runtime directories' space usage depends on user file size, document volume, and other factors, such as file retention schedules.

For more information on NLS, refer to the NLS documentation.

Prerequisites

IS software patch requirements

Make sure all operating system, database, and Service Pack levels follow the IS version guidelines before installing or running SDS_tool. See the SDS_tool readme file for more information.

IS server configuration requirements

Verify the `hosts` file contains the correct ISTK values for your IS domain and HSMA peripherals. An example of your IS domain:

```
Tcpip systemname domainname domainname-FileNet-nch-server

192.168.10.2 psverve verve verve-FileNet-nch-server
```

Note The IS *domainname* is case-sensitive. Specifically, SDS_TOOL_DOM is a two-part *domain:organization* listing in the SDS_tool environment,

and must match the *domainname* entry in the `hosts` file. Keep the `hosts` file synchronized with your SDS_tool environment.

Install Image Services ToolKit (ISTK)

Load the ISTK CD appropriate to your operating system (UNIX or Windows) and follow the program's installation instructions.

Note You must have Image Services installed to run the SDS_tool.

2
Software Installation

You can install SDS_tool on either a UNIX or Windows operating system. Follow the instructions appropriate to your operating system.

SDS_tool for UNIX

This section describes installing SDS_tool on UNIX systems only. (To install SDS_tool on a Windows operating system, see “SDS_tool for Windows” on page 21.)

To change the installation directory, add the parameter where *path* is the directory’s path.

1 Log in as the root user on the server where you are installing SDS_tool.

2 Install SDS_tool into the `/fnsw/local/bin` directory.

3 Insert the CD. Change directory to `/fnsw/local/bin`.

Note On an HP system only, use the following command to mount the CD:

```
>mount -o cdcase /dev/cd0 /cdrom
```

4 Run the SDS_tool installation program using the following command:

```
>csh ./PS_install
```

Note The command `csh` shell (c shell) specification is necessary if you are not using the .csh shell. If already using the .csh shell, `csh` in the command will not cause installation problems.

More than one csh is available on a UNIX system. If the installation fails, issue the `./PS_install` command without specifying a shell.

5 Verify that the following files were written to the install directory (`/fnsw/local/bin`) for AIX, HPUX, Solaris:

```
SDS_tool
libApiDS.a (libApiDS.sl, libApiDS.so)
libFPLibrary.a (libFPLibrary.sl, libFPLibrary.so)
libFPLibrary32.a (libFPLibrary32.sl,
libFPLibrary32.so)
libFPParser32.a (libFPParser32.sl, libFPParser32.so)
libPAI_module32.a (libPAI_module32.sl, libPAI_
module32.so)
PSs (libPSs.sl, libPSs.so)
TSMs (libTSMs.sl, libTSMs.so)
en_US/dsmclientV3.cat
debug.sds
doclist.lst
file.sds
logon.sds
modify.sds
nice.sds
recover.sds
report.sds
sdslist.lst
shell.sds
status.sds
workdir.sds
```

Note After installation, you can move SDS_tool to another directory by moving the files from /fnsw/local/bin.

If running SDS_tool against IBM TotalStorage DR550, the directory `en_US` must exist in the local directory and contain the message file

`dsmclientV3.cat`. If the file does not exist, SDS_tool functions fail with minimal error reporting.

SDS_tool for Windows

This section describes installing SDS_tool in a Windows environment. (To install SDS_tool on a UNIX operating system, see “SDS_tool for UNIX” on page 19.)

1 Log in on the server where you are installing SDS_tool.

2 Insert the CD to launch PS_Install.exe.

3 Select **Continue** at the Application Package Installer dialog box.

4 Find the **Available Releases** frame at the software installation dialog box, and select your ISTK version, then select **Install**.

5 Stamp the module to check the ISTK version level on a Windows server. For example:

```
stamp <drive>:\fnsw\client\shobj\*SysV*
```

An ISTK 4.0 module stamp example:

```
D:\fnsw\lib\shobj\wal_sysv.dll (Windows bin):

system 4.0.5.7(0) (lib, Wed Oct 15 08:38:16 2006)
developer 4.0.0.0.11 (lib, Wed Oct 15 08:38:14 2006)
SubSys: mv, Rel_type: wal_nt, SCR#: 184822, mode:
100666, size: 447160
```

6 Verify the installation location.

Note The initial installation location for SDS_tool is C:\fnsw_loc\bin. This installation location changes to a folder path in the same general loca-

tion as the IS and ISTK software installations. To specify the area as the installation location, select **No**.

7 Specify a new install location when the Destination Path Selection dialog box opens if you selected **No** in Step 5. Select or create the new install location, then select **Accept**.

A typical SDS_tool installation point:

```
< drive letter>: \fnsw_loc\bin
```

CAUTION When selecting an installation path, do not select pathnames that contain spaces. SDS_tool does not work if there are spaces in the filename path.

8 Select **Yes** to verify the new installation location by or select **No** to return to the Destination Path Selection dialog box.

9 Select **Quit** to close the Installation Program interface.

10 Verify that the following files installed:

```
FPLibrary.dll
FPParser.dll
FPToolbox.dll
SDS_TOOL.exe
NetSnmp.dll
PAI_module.dll
PSs.dll
TSMapi.dll
TSMs.dll
dsmntapi.dll
fpos32.dll
tsmLvsaMsg.dll
tsmapi.dll
tsmutil1.dll
debug.sds
```

```
doclist.lst
file.sds
logon.sds
modify.sds
nice.sds
recover.sds
report.sds
sdslist.lst
shell.sds
status.sds
workdir.sds
```

3
Configuring SDS_Tool

Complete the following tasks before executing SDS_tool for the first time:

- Update the system hosts file.
- Verify that SDS_tool communicates with FileNet ISTK Runtime.
- Confirm you can log onto the FileNet system.

“hosts” file entry

Locate the “hosts” file on your server:

- UNIX: /etc/hosts
- Windows: <drive>:\Windows\system32\drivers\etc\hosts

To correctly execute the ISTK logon call, ISTK applications require specific entries in the “hosts” file. (See the “hosts” file entry example below.) The entry is specific to ISTK applications and must be added to the “hosts” file before running a FileNet utility. Enter the FileNet domain name entry exactly as it appears in the FileNet Application Executive; if the name includes capital letters with underscores, enter the name using capital letters and underscores.

The IP address is the first entry, followed by a tab. Enter the server name next, followed by a space. The FileNet domain name is the third entry, followed by another space. Finally, enter the four-part FileNet Network Clearing House (NCH) server name.

Example:

```
# IP address    servername serveralias
domainname-organization-nch-server
183.52.10.11    local_acctng1 acctng1
acctng1-filenet-nch-server
```

Note If you require entries with different domain name case sensitivity in the four-part NCH server name, add two four-part NCH server names to the line entry for your FileNet server.

Environment

Set up the PATH to the install directory so that the environment variables in the user environment call the ISTK shared libraries, not the IS libraries. Any user running SDS_tool must have the path to the ISTK libraries. UNIX users must have a password to access the PATH. For more information, refer to the ISTK install readme file.

The UNIX PATH configuration differs from the Windows configuration; do not confuse the two configurations. Also, do not limit the PATH to the examples below, as there are other software and OS requirements in each environment PATH you must retain.

Rather than using a configuration file, SDS_tool first retrieves the configuration from its environment variables. This includes the:

- IS user SDS_Tool_Admin's password.
- IS two-part domain name, SDS device type.
- Debug level.
- Output file name.

When SDS_tool starts, it prompts for any variables it cannot find in the environment.

SDS_tool uses the following environment variables. Set these variables prior to running SDS_tool, or simply respond to the SDS_tool prompts.

- SDS_TOOL_PW SDS_Tool_Admin Password
- SDS_TOOL_DOM FileNet Two-part NCH Domain
- SDS_TOOL_WD SDS Device Working Directory
- SDS_TOOL_DEV SDS Device Type
- SDS_TOOL_ADDR SDS Device IP Address (if applicable)
- SDS_TOOL_OUT output file
- SDS_TOOL_DEBUG debug level (optional)
- SDS_TOOL_NICE delay between batches of 100 (optional)

Verify that the environment variables are set by running the following command.

For UNIX:

```
env |grep SDS_TOOL
```

The output displays:

```
SDS_TOOL_PW=SysAdmin
SDS_TOOL_DOM=psmud:FileNet
SDS_TOOL_WD=/NAS/NLStest
SDS_TOOL_DEV=Centera
```

```
SDS_TOOL_ADDR=10.10.16.1
SDS_TOOL_OUT=outfile0
SDS_TOOL_DEBUG=0
SDS_TOOL_NICE=0
```

For Windows:

```
set | findstr SDS_TOOL
```

The output displays:

```
SDS_TOOL_PW=SysAdmin
SDS_TOOL_DOM=pschili:FileNet
SDS_TOOL_WD=\\SnapLock1\NLS_test
SDS_TOOL_DEV=SnapLock
SDS_TOOL_OUT=outfile0
SDS_TOOL_DEBUG=0x00
SDS_TOOL_NICE=0
```

The next example shows a typical run where SDS_tool gets its configuration from user input. The bold text is written by SDS_tool and the plain text is user input:

Enter Password:

Enter two-part Domain: psmud

Invalid domain name. Should be 'Domain:Org'

Enter two-part Domain: psmud:FileNet

Enter Work Directory: /NAS/NLStest

Enter SDS device: Centera

Enter SDS address: 10.10.16.1

```
Enter SDS output file: sds_tool.out

SDS debug: 0x00
```

ISTK file ownership and permissions on UNIX

ISTK provides an installation readme outlining UNIX environment permissions and ownership settings on the ISTK modules. Follow the readme to correctly access the ISTK modules during application runtime. Most ISTK modules should be owned by fnsw:fnusr. Two modules, wal_purge and wal_daemon, must be owned by root:fnusr. For complete instructions, refer to the ISTK readme on the P8 Image Services ToolKit CD.

Note Due to Windows limitations, IS only allows one IS user to log in at a time. If you have NLS or another ISTK-using application running, you cannot run SDS_tool unless those applications are accessing IS as the SDS_Tool_Admin user. You can only logon to SDS_tool as SDS_Tool_Admin.

User environment

All platforms

On all systems that SDS_tool logs on to, create the user SDS_Tool_Admin in Xapex Security Administration, with full membership in the SysAdminG group as its primary and session group. When creating or updating the SDS_Tool_Admin user, check the Supervisor flag.

This gives SDS_tool permission to operate in IS. Note that the password that you assign to SDS_Tool_Admin is used to start SDS_tool.

FileNet security user names are case-sensitive, so create the user exactly as stated.

Note When creating or updating the SDS_Tool_Admin user in Xapex, turn on the Supervisor flag.

UNIX operating system user only

Although the fnsw user is most common, SDS_tool supports different UNIX users. Give the user read, write, and execute privileges for all directories, files, and programs that SDS_tool uses. The fnusr, fnop, fnadmin, and dba groups must include the UNIX user as a member.

The UNIX environment assigns an ID number to all users and groups. The IS and protected storage devices must both contain the user and group that run SDS_tool. Furthermore, both IS and protected storage device users and groups need the same ID number to set archived document privileges correctly.

Windows and UNIX IS user

Create an IS user with read, write, and execute privileges for documents on the SDS device.

SDS_tool operation

This section describes SDS_tool operation in the SDS solution. SDS_tool manages documents from FileNet's Image Services on protected storage device libraries.

Interactive execution

To start SDS_tool, use SDS_tool (UNIX) or SDS_tool.exe (Windows). When SDS_tool starts, it writes out the current environment:

```
FileNet Domain    : lbush:FileNet
Work Directory     : s:\NLS_test
SDS device          : SnapLock
SDS output file: lbush
SDS debug     : 0x00
```

Type `Help` at the command prompt to bring up the command list:

```
shell (!)
debug
file
logon
modify
nice
recover
report
script
status
workdir
```

For detailed help, follow `Help` with a command name, such as `help report`.

4
Using SDS_Tool Commands

To use SDS_tool, use the commands in this chapter.

Shell commands

Like other IS tools, SDS_tool uses an exclamation point as an escape to the default shell.

Function: To run shell commands from within the SDS_tool.

Device dependence: None.

Restrictions: Within SDS_tool, some IS Tools cannot be invoked.

Example: Use a single command such as `! Dir o/d` or invoke a new shell for entering commands, such as `! Cmd` (Windows) or `! ksh` (UNIX).

Debug

Function: The user-set `debug` level interactively controls the debug information stored in the **sdst**yyyymmdd log file.

Device dependence: None.

Restrictions: None.

Example: To see the current debug level, type **debug** by itself. The current debug level, such as **debug** `0x10`, displays.

You can also use `debug` in conjunction with using SDS_tool to run a script.

```
SDS_Tool -debug<debug_level> <script_name>
```

Debug levels include:

- **0x00:** The default is that no debug information is stored.
- **0x01:** The information level emits general, high level information, such as entering a routine or other basic information.
- **0x02:** The debug level releases detailed information, including data values and programmer information.
- **0x04:** Similar to other utilities, the spy mode gives information about ISTK calls and their related data.
- **0x08:** The memory mode gives information about memory allocation and de-allocation, useful for detecting memory leaks.
- **0x10:** The SDS filesystem calls give information about actions to a file-system-based SDS, reads, writes and file actions.
- **0x20:** The SDS API calls yield information about third-party API calls to the SDS devices. You can logically combine the debug level modes or use multiple levels simultaneously.
- **0x2f**: Sets all debug flags.

File

Function: The interactive `file` command sets the output file name that receives the tool data output, except for logging information. **file** uses either a simple file name, such as `sds_data.out`; a relative

path, such as `data/sds_data.out`; or a fully qualified path name, such as `/fnsw/local/bin/sds_data.out`.

Device dependence: None.

Restrictions: None.

Example:

```
file <filename>
```

Logon

Function: SDS_tool initially logs on to the IS domain specified in the environment, or through the interactive start up dialog box. To change log on without exiting the tool, use the command format:

```
logon  domain01:Filenet Tivoli 10.10.84.10
logon domain02:FileNet Centera 10.10.85.1
logon domain03:FileNet SunSAR
logon domain04:FileNet SnapLock
```

Note Network Appliance SnapLock and Solaris SunSAR do not need IP addresses because they are simple file systems. But SDS_tool needs the IP address to talk to IBM TotalStorage DR550 and to EMC Centera.

The domain and organization are the FileNet two-part domain name.

Device dependence: The device is the SDS device type: Network Appliance SnapLock, EMC Centera, Solaris SunSAR or IBM TotalStorage DR550. The device address is the device's IP address, required only for API devices.

Restrictions: Device address only appears for IBM TotalStorage DR550 and EMC Centera. If using other utilities, you must log on as SDS_tool admin (Windows only).

Example:

```
logon <domain:organization> <device> [<device_address>]
```

Logoff is automatic.

SDS_tool uses the SDS_Tool_Admin user for logon security. You must have privileges to change the FileNet database.

Modify

Function: Use the `modify` command to Set or Extend the retention time or to delete an SDS object. If a delete or archive date is set in the Index database, SDS_tool uses that as the retention date. If the date is undefined, the user specified date applies.

The `help modify` command outlines the syntax:

modify <id_type>={+}<id_value> <option>=<option_value>

Where:

- **id_type** is the keyword docid or sdsid.
- **id_value** is the actual Document ID, SDS ID or a file list of IDs depending on the keyword used. The SDS ID is the SDS object ID stored in the FNP_CLIPID field of the IS Index database.
- **option** is Set, Extend, or Delete.

- **option_value** is:

  Set = mm/dd/yyyy - Set new retention to specified date.

  Set = Hold - Hold indefinitely until EBR event.

  Set = Release - Removes the Hold that has been applied.

  Set = Ebr - Change hold to EBR.

  Extend =+n{d | m | y} - Add 'n' units (day, month, year) to retention

  Delete - Set object for deletion if retention has expired.

Note Only one **option** is valid at a time.

Examples:

`docid=100000 Set=12/31/2008`
Sets the retention of document 100000 to 12/31/2008.

`docid=100000-105000 Set=12/31/2008`
Sets the retention of documents in the range from 100000 to 105000 to 12/31/2008. Range can also be entered as: docid=100000,105000.

`docid=+docidlist Set=12/31/2008`
Sets the retention of the Document IDs in the docidlist file to 12/31/2008. This file can contain docids, one per line, or can contain ranges of docids, one range per line.

`sdsid=41J3CC8A087Eve2AC1813NAJU1B Set=12/31/2008`
Sets the retention of the SDS object ID to 12/31/2008.

`sdsid=+sdsidlist Set=12/31/2008`
Sets the retention of the SDS object IDs in sdsidlist to 12/31/2008.

`docid=100000 Set=hold`
Sets the document 100000 to a Hold state.

`docid=100000 Set=release`
Removes the Hold on document 100000.

In file-system-based SDS devices, retention is the SDS object access time. Use the UNIX command `ls -lu` or the Windows command `DIR/TA` to see the time. If the retention is undefined in Image Services, SDS_tool extends the retention time by resetting the object access time. It cannot shorten the retention time.

When setting retention with the `Set` or `Extend` commands, SDS_tool first checks the Index Database for a retention date. If the requested retention is less than the current retention date, SDS_tool does nothing. If the date is less than the requested one, it extends the retention. If the retention is not defined in the Index database, SDS_tool uses the requested retention. SDS_tool cannot shorten or remove an established retention period.

Similarly, SDS_tool checks the Index Database retention before deleting any object. To delete, retention must expire on the object according to both the FileNet and SDS devices.

Device dependence: The modify command has the following device dependencies and restrictions:

- modify docid=100000 **set=ebr**

- On Network Appliance's SnapLock or Solaris SunSAR, **EBR** has no effect. This is because these devices are not EBR-enabled.
- If EMC Centera is EBR-enabled and f_deletedate is set in FileNet, it sets retention to the f_deletedate; otherwise, no effect.
- On IBM TotalStorage DR550, if the Management Class (MC) is set to enforce retention, it sets retention according the MC definition.

- modify docid=100000 **set**=**hold** - Hold applies only to IBM TotalStorage DR550 and EMC Centera.
  - On Network Appliance's SnapLock or Solaris SunSAR, **Hold** has no effect.
  - On EMC Centera and IBM TotalStorage DR550, it will hold the object indefinitely until the hold is released, regardless of any other retention settings.
  - IBM Total-Storage DR550 objects are assigned by NLS_ Archive to the associated management class that is set with the appropriate retention rules. All IBM TotalStorage DR550 objects are set to “hold” if an expiration is not specified
  - For EMC Centera, “hold” means it is being retained indefinitely until you determine an expiration date.
- modify docid=100000 **set=release**
  - On Network Appliance's SnapLock or Solaris SunSAR, **release** has no effect.

- On EMC Centera and IBM TotalStorage DR550, **release** removes the hold applied to the SDS objects.

- modify docid=100000 **set**=**mm/dd/yyyy**
  - On Network Appliance's SnapLock or Solaris SunSAR, if f_deletedate is not set, it sets retention to the date provided otherwise date is ignored
  - On EMC Centera, if f_deletedate is not set, it sets retention to the date, provided the object is not EBR-enabled. If the object is EBR-enabled, the object can only be modified to a date beyond the EBR date.
  - On IBM TotalStorage DR550, retention date is controlled by the management class and cannot be modified.
- modify docid=100000 **extend**=**+n(dlmly)**
  - On Network Appliance's SnapLock or Solaris SunSAR, if f_deletedate is not set, it extends retention by the given units. If f_deletedate is set, the command is ignored.
  - On EMC Centera, if f_deletedate is not set, it extends retention by the given units provided the object is not EBR-enabled. If f_deletedate is set, the command is ignored.
  - On IBM TotalStorage DR550, **extend** has no effect because retention date is controlled by the management class.
- modify docid=100000 **delete**
  - The document must be closed and expiration must be met.
  - The document is removed from the SDS device and the DIR is deleted.

Nice

Function: The `nice` command delays the execution a long list of operations by “sleeping” a specified number of seconds between each batch. Currently, a batch is 100 operations; that is, SDS_tool reports on or modifies 100 documents, then sleeps.

Device dependence: None.

Restrictions: None.

Example:

Type

```
nice
```

by itself for SDS_tool to return the current nice value:

```
nice = 5 seconds
```

Or, type nice <seconds>, such as

```
nice 10
```

for SDS_tool to set a new nice value, 10 seconds. For example, with 10 seconds, when a long list or range of documents processes, SDS_tool performs 100 operations, prints out

```
Processed 100 documents
```

sleeps for 10 seconds, then processes another 100 documents. This sleep or inactivity period allows the system to catch up with other processing.

Recover

Function: The `recover` command restores an SDS object or list of objects to the FileNet IS. If the Document Index Record (DIR) is missing, SDS_tool restores it from the object DIR record. If the DocLocator table entry is missing, SDS_tool re-commits the document. The recover command syntax is:

```
recover <id_type>={+}<id_value>
```

Where:

id_type is the keyword *docid* or *sdsid*.

id_value is the actual Document ID, SDS ID or a file list of ID's depending on the keyword used.

Examples:

```
recover docid=100000
recover docid=+mydoclist
recover docid=100000-120000 (doc id range)
```

In this case, the SDS ID allows SDS_tool to access the SDS device "BLOb," a file containing a single document and FileNet metadata. The SDS device repository contains enough data to recover FileNet documents to the most recent archived document version. However, the SDS device might not capture updates to document security and index values, or annotations and other values. SDS_tool automatically attempts to recover all documents that the SDS query lists.

The recover function takes a single ID or a list of document IDs for recovery. The recovery routine reads the document information from the object file on the SDS repository. If the document index record does not exist, SDS_tool recommits the document back into Image

Services. When there is an SDS object, SDS_tool takes the DIR embedded in the object and restores the DIR table on Image Services. If the document is missing from the FileNet Document Locator table in the permanent database, SDS_tool creates a batch and recommits the document.

Caution For Centera and IBM TotalStorage DR550, if documents were archived without the document image file on the NLS Working Directory, you must use SDSID to recover. This includes cases where “DEL” was used in the SdsDisposition setting in NLS.cfg.

Note After recommittal, a document might not be in the cache, but it is retrievable.

Report

Function: The `report` command gives detailed information about the IS document and/or the SDS object, and checks their validity. The syntax is:

```
report <id_type>={+}<id_value> {option=FileNet|SDS|both}
{file=<filename>}
```

Where:

id_type is the keyword *docid* or *sdsid*.

id_value is the actual Document ID, SDS ID or a file list of IDs depending on the keyword used.

option is Verify

option_value is:

Filenet = Document Index Record is verified

SDS = Document location is verified

Both = Both Document Index Record and Document location is verified

file is used to redirect output

Example of a typical report on a single document:

```
<SDS_tool>report docid=101145
-----------------------------------------------
------------------------------
INX Info: Id=101145
In Storage Library   SDS Info: Device  =SnapLock,
DevInfo =None required
FS Info: \\snaplock1\NLS_test\NLS_
test\1100123456\00\00\0B\B8\00\01\8B\101145
Creation Time : Wed Apr 18 13:33:32 2007
Expiration Time: Wed Apr 18 13:57:01 2007
Valid InxDB, PermDB, SDS, Closed=F, Action=None
```

Note f_deletedate and f_retent date only display if they are not NULL. Otherwise, they will not display.

Script

Function: The `script` option specifies a file containing one or more SDS_tool commands to execute sequentially.

Device dependence: None.

Restrictions: None.

Example: To run a script, type the following at the command prompt:

```
SDS_tool -c <script_name>
```

See **Chapter 5, "Typical Scripts," on page 48** for examples.

Status

Function: The `status` command adds the SDS device status information to the log file. The status varies among SDS devices.

Device dependence: You get a different status depending on the device type.

Restrictions: None.

Examples:

For example, an IBM TotalStorage DR550 device's status:

```
<SDS_tool> status

User name              : SDS_Tool_Admin

FileNet Domain         : nt44:FileNet

Work Directory         : \\nfsdepot04\nt44\NLS_FS

SDS device             : IBM TotalStorage DR550

SDS address            : 10.10.8.35

SDS output file        : outfile

SDS debug              : 0x00
```

```
<SDS_tool>
```

An EMC Centera device example:

```
<SDS_tool> status
User name              : SDS_Tool_Admin
FileNet Domain         : sun01d2r:FileNet
Work Directory         : /fnsw/msar/SDS_NLS
SDS device             : Centera
SDS address            : 10.10.16.1
SDS output file        : outfile
SDS debug              : 0x00
<SDS_tool>
```

A Network Appliance SnapLock device example:

```
 SnapLock Volume Information:
    Host = 'SnapLock1'
    Vol  = 'NLS_test'
    UUID = '0x3de27d60-11d82b59-a000b1b2-c1df0098'
    Path = '/SnapLock1/NLS_test'
```

A Solaris SunSAR device example:

```
<SDS_tool> status
User name              : SDS_Tool_Admin
FileNet Domain         : ibm23d2r:FileNet
```

```
Work Directory          : /fnsw/SDS_SunSAR
SDS device              : SunSAR
SDS address             : None required
SDS output file         : outfile
SDS debug               : 0x00
<SDS_tool>
```

Workdir

Function: The `workdir` command changes the current working directory to the requested one, providing SDS_tool can write to the requested directory.

Device dependence: None.

Restrictions: None.

Example:

```
workdir <path>
```

Log files

General: Files that contain status, error and debugging information.

The SDS_tool errors appear in three places, depending on the error type:

- The SDS_tool log, sdst<yyyymmdd>, always located in the directory SDS_tool runs in. If enabled, look here for SDS_tool errors and debug information.
- The IS error log file.

- The ISTK error log file for the current date.

5

Typical Scripts

The SDS_tool release package includes sample scripts with slightly different versions for Windows and UNIX. The scripts have representative syntax for each command type. When writing a script, SDS_tool checks for the password variable. If it doesn't find it, SDS_tool prompts for the password.

The basic syntax for running an SDS_tool script is:

```
SDS_tool -c <script_name>
```

Use a simple script to report on a single Doc ID:

```
report docid=187430 verify=both
quit
```

The command output is:

```
ibm23d2r(fnsw)/fnsw/local/bin> ./SDS_tool -c one_
doc.sds

SDS_tool: starting

Logging on with the hard-coded FileNet user 'SDS_Tool_
Admin'

FileNet Domain    : ibm23d2r:FileNet

Work Directory    : /fnsw/msar_cag/SDS_NLS

SDS device        : Centera

SDS address       : 10.10.16.1

SDS output file : outfile
```

```
SDS debug          : 0x00

System Shell       : /bin/ksh

Logon 'SDS_Tool_Admin' to 'ibm23d2r:FileNet' suc-
cessful

-------------------------------------------------------
-----

INX Info: Id=187430

In Storage Library SDS Info: Device=Centera,
DevInfo=10.10.16.1

DevSdsId=5EEE9EG1CSDOJeB3A9BGUAHEPM5

InxSdsId=5EEE9EG1CSDOJeB3A9BGUAHEPM5

FsSdsId=

FS Info: /fnsw/msar_cag/SDS_NLS/10396/00/00/0C/32/00/
02/DC/187430

Creation Time  : Wed Apr  4 19:45:47 2007

Expiration Time: Wed Apr  4 19:50:47 2007

Valid InxDB, PermDB, SDS, Closed=F, Action=None

SDS_tool: done
```

Note For an explanation of report data, see “Report” on page 42.

The sample script sets the Doc ID retention to a given date. The `file` command sends the scripts output to the specified file; in this case, `sds01.out`.

```
file sds01.out

modify docid=147002579 set=01/01/2008

quit
```

Note Retention is set only if the current retention is undefined in the FileNet IS; otherwise, the object retention is set to the FileNet Delete or Archive date.

This script extends the Doc ID retention by six months:

```
file sds02.out
report docid=150000005
modify docid=150000005 extend=+6m
quit
```

This script deletes a document, provided the document's retention has expired:

```
file sds03.out
modify docid=150000600 delete
quit
```

Note When using the keywords `sdsid` or `docid`, the corresponding value is a single ID, a file containing a list of values, and a range of doc IDs. For example:

```
modify docid=150000600-150000999 delete
modify sdsid=+sds_id_list set=01/31/2008
modify docid=+doc_id_list extend=+1d
modify docid=100000-199999 extend=+1m
```

Released sample scripts

The SDS_tool software package contains sample scripts installed in the same directory as the executables. The scripts echo what they are doing to the screen as they run. You can invoke the scripts by using the `-c` option. For example:

```
SDS_tool -c debug.sds
```

Some scripts work as released, but those requiring site-specific data, document IDs and domain names must be modified by the user.

shell.sds

Function: The sample script `shell.sds` contains commands that demonstrate how to execute shell commands inside the script.

Device dependence: None.

Restrictions: None.

Example:

For UNIX:

```
# Set output file
file shell.out


# Any shell commands can be executed after the
exclamation point
! echo
! echo
```

```
! echo "# Any shell commands can be executed after the
exclamation point"

! echo ls

! ls


# SDS_tool uses the default shell on UNIX systems

! echo

! echo

! echo  "# SDS_tool uses the default shell on UNIX
systems"

! env |grep SHELL

! echo


quit
```

For Windows:

```
# Set output file
file shell.out

# Any shell commands can be executed after the
exclamation point

! dir

# SDS_tool uses the DOS shell on Windows systems

! set |findstr OS=
```

debug.sds

Function: The sample script `debug.sds` contains commands demonstrating how to set the SDS_tool debug level.

Device dependence: None.

Restrictions: None.

Example:

```
# Display the current debug level
! echo
! echo
! echo "# Display the current debug level"
! echo
debug


# Set the debug level to maximum
! echo
! echo
! echo "# Set the debug level to maximum"
! echo
! echo debug 0x2f
debug 0x2f


! echo
```

```
! echo

! echo "# Display the current debug level"

! echo

debug

! echo


quit
```

file.sds

Function: The sample script `file.sds` contains commands that change the output file.

Device dependence: None.

Restrictions: The Doc ID must be modified by the user for the script to work.

Example:

```
# Change the output file at any time

! echo

! echo

! echo "Output file is file1.out"

file file1.out

report docid=100000


! echo
```

```
! echo
! echo "Output file is file22out"
file file2.out
report docid=100000

! echo
! echo
! echo "Two output files"
! ls -l file?.out

quit
```

logon.sds

Function: The sample script `logon.sds` contains commands that you log on to another FileNet domain.

Device dependence: None.

Restrictions: You must put in your own FileNet domain name and device name.

Example:

```
# Logons can change dynamically

! echo
! echo
```

```
! echo "# Logon to another domain"
logon pstrain:FileNet

! echo
! echo
! echo "# Logon to another domain and device"
! echo "logon psmb20:FileNet SnapLock"
logon psmb20:FileNet SnapLock
quit
```

modify.sds

Function: The sample script `modify.sds` contains commands that modify or delete a single document.

Device dependence: None.

Restrictions: You must change the Doc ID to a valid Doc ID.

Example:

```
# Modify document retention or delete document

! echo
! echo
! echo "# Modify document retention -- set to a specific
date"
! echo "modify docid=100000 set=1/1/2010"
```

```
modify docid=100000 set=1/1/2010


! echo

! echo

! echo "# Modify document retention -- extend 1 day from
current date"

! echo "modify docid=100000 extend=+1d"

modify docid=100000 extend=+1d


! echo

! echo

! echo "# Modify document retention -- extend 1 month from
current date"

! echo "modify docid=100000 extend=+1m"

modify docid=100000 extend=+1m


! echo

! echo

! echo "# Delete document retention if retention expired"

! echo "modify docid=100000 delete

modify docid=100000 delete
```

nice.sds

Function: The sample script `nice.sds` contains commands that set the number of seconds SDS_tool pauses between sets of documents.

Device dependence: None.

Restrictions: None.

Example:

```
# Nice makes SDS_tool pauses between sets of documents


! echo

! echo

! echo "# Nice by itself shows the current nice interval"

! echo "nice"

nice


! echo

! echo

! echo "# Nice and a number of seconds sets the nice
interval"

! echo "nice 120"

nice 120


nice
```

```
quit
```

recover.sds

Function: The sample script `recover.sds` contains commands that recover a single document with missing data on the FileNet system.

Device dependence: None.

Restrictions: You must enter a legitimate Doc ID. If the document does not need recovery, it will say so.

Example:

```
# Recover restores a document in FileNet from the SDS
Device


! echo

! echo

! echo "# Recover document using Document ID"

! echo "recover docid=100000"

recover docid=100000


! echo

! echo

! echo "# Recover document using a list of SDS ID's"

! echo "recover sdsid=+sdslist.lst"
```

```
recover sdsid=+sdslist.lst


quit
```

report.sds

Function: The sample script `report.sds` contains commands that create a report on a single document, or range of documents in different styles.

Device dependence: None.

Restrictions: You must modify the Doc ID or SDS ID.

Example:

```
# Set the out file
file report.out


# Report by FileNet Document ID ------------------------
----------------


# FileNet report only
report docid=100000 verify=FileNet


# SDS Device report only
report docid=100000 verify=SDS
```

```
# Both FileNet and SDS Device report

report docid=100000 verify=both


# Default is 'both'

report docid=100000


# Range of Document IDs

report docid=100000-200000


# List of Document IDs

report docid=+doclist.lst


# Report by SDS ID ------------------------------------
----------------


   # FileNet report by IBM TotalStorage DR550 style SDS
ID

report sdsid=0:5200319:/00/00/2E/EE/00/C7/E4/13100142.0_
     0.blob verify=FileNet


# SDS report by Centera style SDS ID

report sdsid=07R7PD73E74BNe0LKHVF1TG3KVS verify=SDS


# Both FileNet and SDS Device report
```

```
report sdsid=0:5200319:/00/00/2E/EE/00/C7/E4/13100142.0_
     0.blob verify=both


# Default is 'both'

report sdsid=07R7PD73E74BNe0LKHVF1TG3KVS


quit
```

status.sds

Function: The sample script `status.sds` contains commands that return the connected SDS device status.

Device dependence: None.

Restrictions: Depending on your device, your report may look different.

Example:

```
# Status puts of the status of the SDS Device in the log


! echo

! echo

! echo "# Status puts the status of the SDS Device in the
log"

status
```

workdir.sds

Function: The sample script `workdir.sds` contains commands that change the current working directory.

Device dependence: None.

Restrictions: You must put this into an existing directory on your system.

Example:

```
# Display the working directory
! echo
! echo
! echo "# Display the current working directory"
! echo
workdir


# Change the working directory
! echo
! echo
! echo "# Change the working directory"
! echo
! echo workdir /tmp
workdir /tmp
quit
```

6
Glossary

Access Node – EMC Centera access port represented by a single IP address.

API – Application Programming Interface. A set of library calls compiled and linked with available libraries to make service requests to another application or system.

Clip ID – Address the application uses to get content from the SDS device. This applies to API-based devices, not file system devices. The term is derived from the EMC Centera C-Clip ID.

Creation time – SDS_tool calls the time an object was originally created “creation time.”

DIR – Document Index Record. A FileNet Index database record that contains index information for a single document, including retention information and the SDS ID, if applicable.

Deletion hold – An indefinite hold that prevents an SDS object’s deletion until the hold is removed. Used by some SDS devices.

Document ID – FileNet 32-bit document identifier number ranging from 100,000 to 3,999,999,999.

Domain – The FileNet Image Services Network Clearinghouse two-part name, e.g. Server1:FileNet.

Event-based retention – An SDS object is initially set to an indefinite retention that is set to a specific period based on a future, external event. For example, executing SDS_tool's **EBR** parameter.

Expiration time – SDS_tool calls the time an object's retention expires the "expiration time."

IS –FileNet Image Services system for document mass storage, retrieval and maintenance.

ISTK – Image Services Toolkit (formerly called WAL or WorkFlo Application Library), a FileNet software package consisting of a set of libraries and other software for remotely interacting with an IS system.

Index Database – A relational database where the FileNet IS stores metadata for a document which includes the SDS ID, dates, class information and user-defined indexes.

Permanent Database – A non-relational, FileNet-propriety database containing the document location information, the surface the document is on, the document's status, and annotations.

Protected Storage – A device with its own internal mechanism for setting an SDS object retention.

Retention – The time period a document or object is retained by an SDS storage device. This period is zero, undefined, a specified time period, an infinite period or an indefinite time period or hold. Hold is when an event-based retention period comes into operation, where the object is retained indefinitely until an external event occurs, such as closing an account, setting the retention to a specified value. The SDS device prevents object deletion until the retention period expires.

SDK – Software Development Kit or a set of APIs, programming tools, libraries, provided by a manufacturer to facilitate third-party software development.

SDS – Single Document Storage. Specific to FileNet, this means documents are stored as individual objects on a peripheral disk array along with their metadata. This is opposed to MSAR or optical disk storage, where many documents are stored in a large binary file.

SDS ID – A string of characters the SDS device uses as a unique identifier for an object version. The objects on file-system-based SDS devices might not have SDS IDs.

SDS Object – A binary file residing on an SDS device that contains the document's data and metadata. The metadata partly consists of the FileNet Index information available at the time the object was written.

UUID – Universally Unique Identifier, typically a 128-bit number that uniquely identifies an object on a network.

Working Directory – Filesystem path or location where SDS_tool looks for SDS objects.